# Apple Bread Problem

**Name: Lavanis A**
**Roll No.: bmat2353**

- The process of simulation is by choosing a random real number uniformly between (0,1).

- If the real number is less than 0.25, $S_n$ moves to its right by 1 unit from $S_{n-1}$.

- If the real number is less than 0.5 but more than 0.25, then $S_n$ moves to its left by 1 unit from $S_{n-1}$.

- If the real number is less than 0.75 but more than 0.5, then $S_n$ moves upward by 1 unit from $S_{n-1}$.

- If the real number is more than 0.75, then $S_n$ moves downward by 1 unit from $S_{n-1}$.

- For the subparts (a) and (b) we create a function named `apple_bread(n,x,y)`, which inputs the number of times the process should occur (n), the x-coordinate of the initial point (x) and y-coordinate of the initial point (y). We implement the flow using the for loop and a set of if-else conditions.

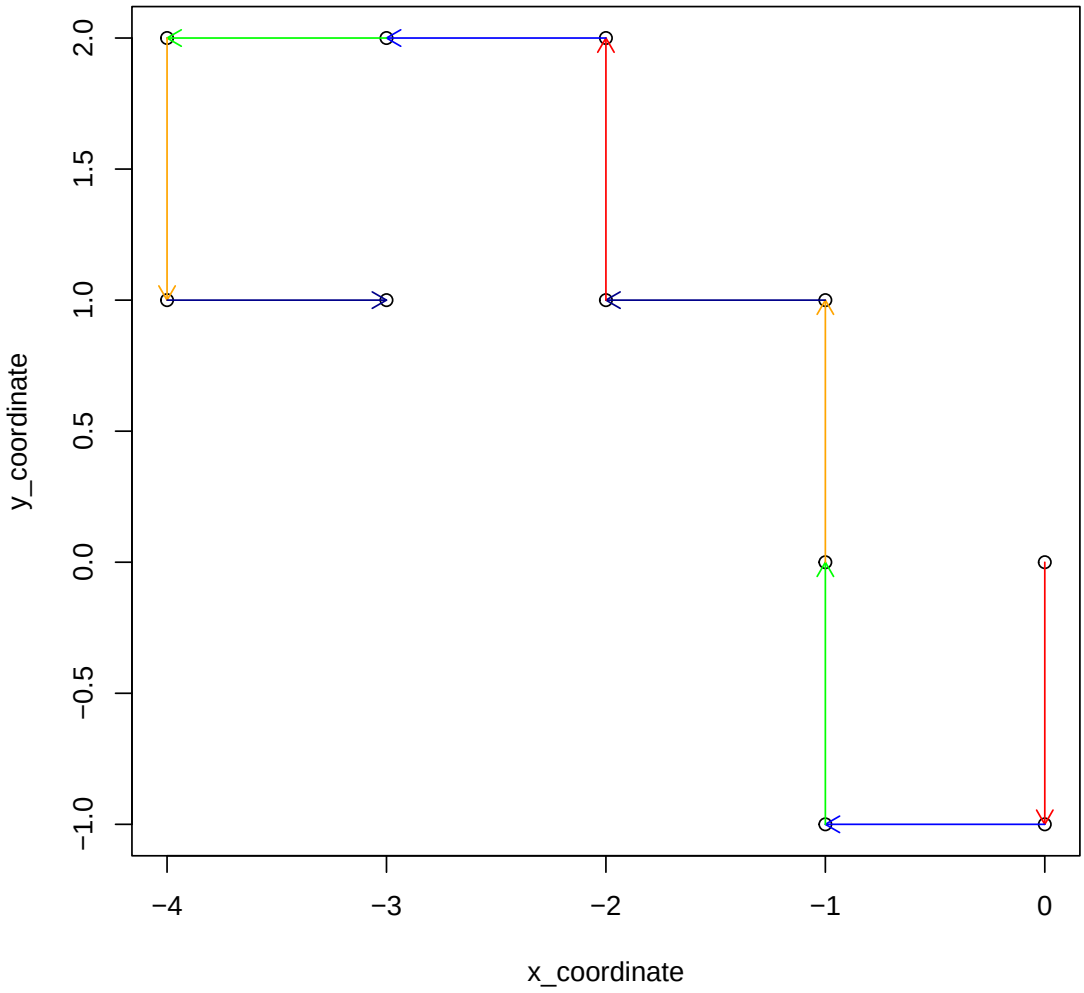
- For the subpart (c) we modify `apple_bread(n,x,y)` to `apple_bread_c(x,y)`, which includes another if condition along with every initial if-else condition of check if the new coordinate is at distance 55 from the origin. Since we do not know the number of times the loop has to run, we replace the for loop with a while loop.

- For the subpart (d) we modify `apple_bread(n,x,y)` to `apple_bread_d(x,y)`, which includes another if condition along with every initial if-else condition of check if the new x and y coordinates are both equal to 0. Since we do not know the number of times the loop has to run, we replace the for loop with a while loop.

```r
#(a),(b)
apple_bread = function(n, x, y){
  x_coordinate = c(x)
  y_coordinate = c(y)
  z = c(0,0)
  for (i in 1:(n)){
    a = runif(1)
    if (a < 0.25){
      x_coordinate[i+1] = x_coordinate[i] + 1
      y_coordinate[i+1] = y_coordinate[i]
    }
    else if(a > 0.25 & a < 0.5){
      x_coordinate[i+1] = x_coordinate[i] - 1
      y_coordinate[i+1] = y_coordinate[i]
    }
    else if(a > 0.5 & a < 0.75){
      x_coordinate[i+1] = x_coordinate[i]
      y_coordinate[i+1] = y_coordinate[i] + 1
    }
    else{
      x_coordinate[i+1] = x_coordinate[i]
      y_coordinate[i+1] = y_coordinate[i] - 1
    }
  }
  plot(x_coordinate,y_coordinate)
  z = 1:(length(x_coordinate)-1)
  arrows(x_coordinate[z], y_coordinate[z], x_coordinate[z+1],
         y_coordinate[z+1],
         angle = 30, length = 0.1,
         col = c("red", "blue", "green", "orange", "darkblue"))
  l1 = length(x_coordinate)
  print(c("The flow terminates at:"))
  print(c(x_coordinate[l1], y_coordinate[l1]))

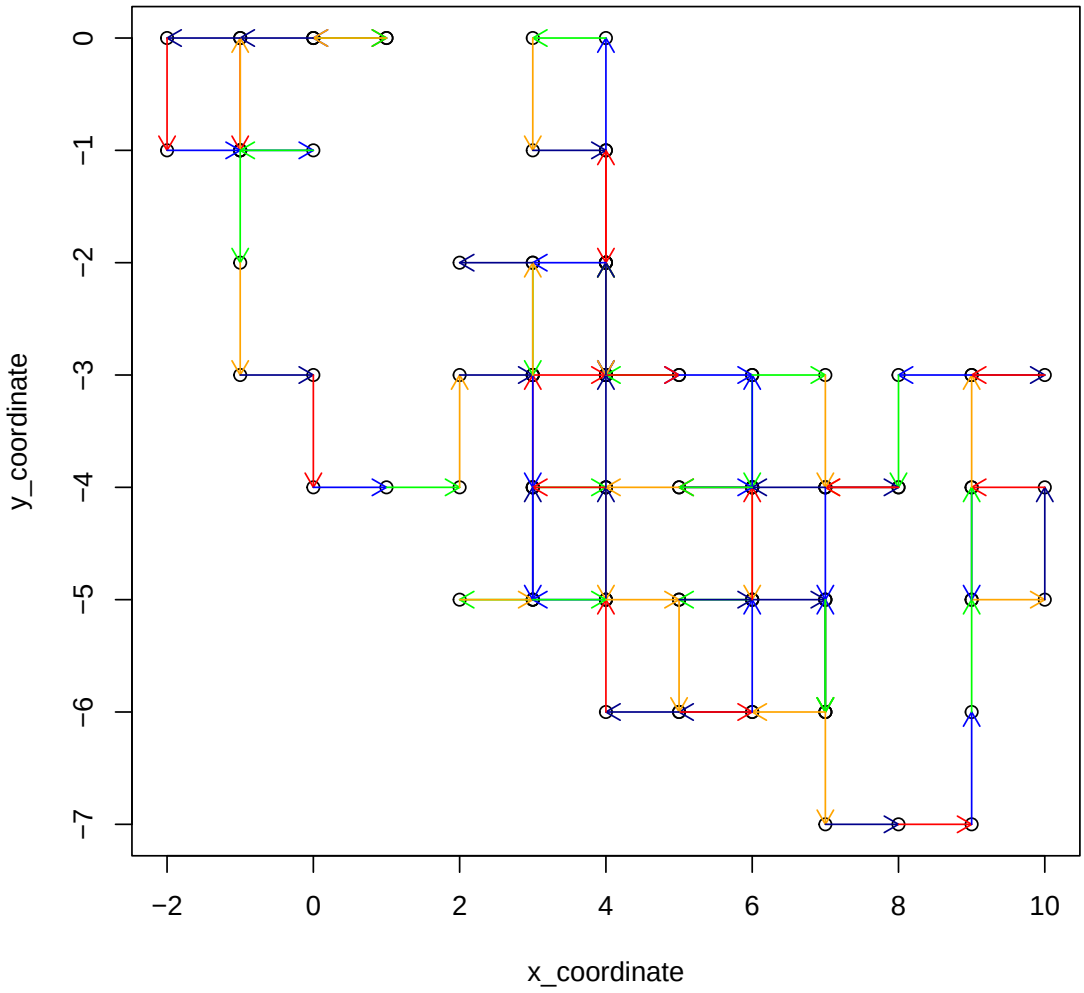
}
```

```
set.seed(2024)
apple_bread(10,0,0)
```




```
## [1] "The flow terminates at:"
## [1] -3  1
```

```
set.seed(4000)
apple_bread(100,0,0)
```




```
## [1] "The flow terminates at:"
## [1]  2 -2
```

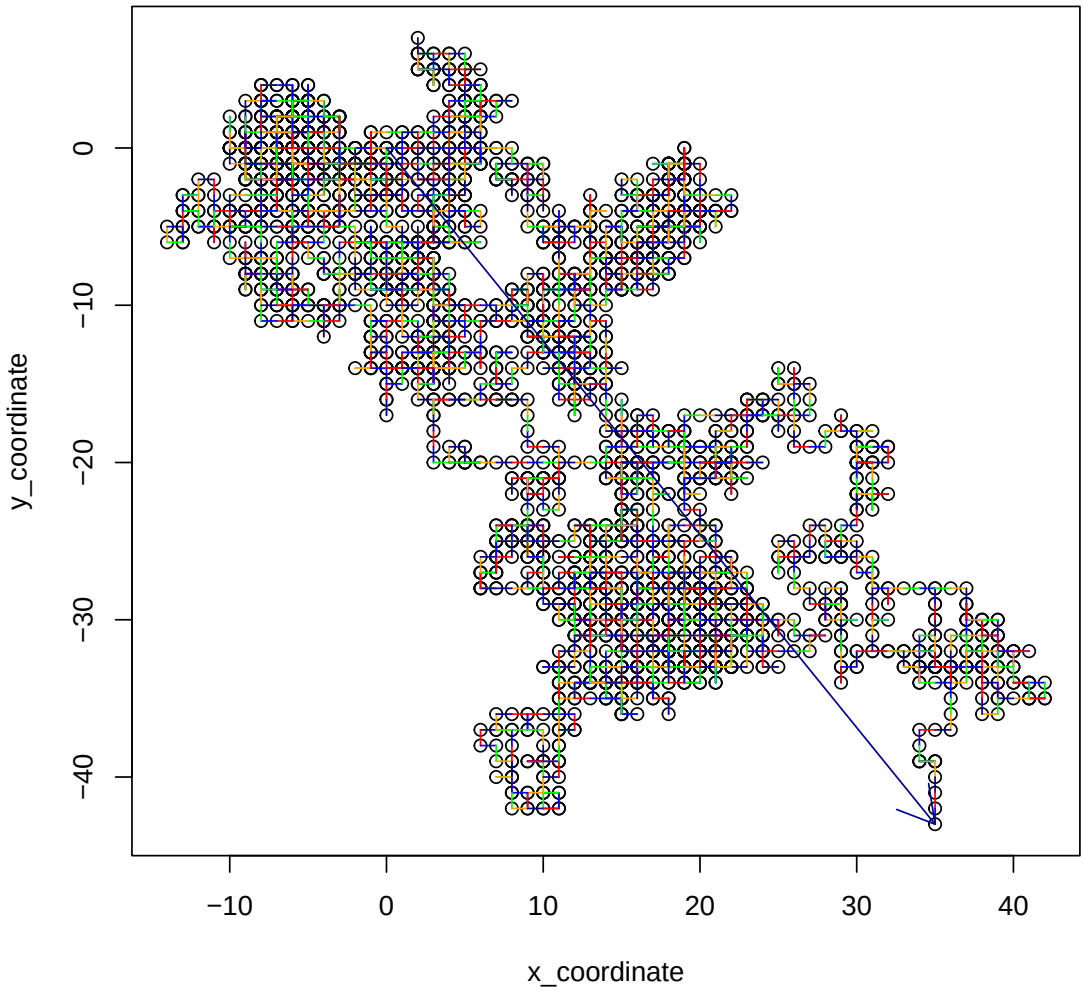
The way one can read the flow of this graph by carefully following the flow of colours (preferrably the arrow tips) as: red, blue, green, orange, darkblue, red, blue, green and so on. The graph starts at (0,0) and ends at (2,-2).

```r
#(c)
set.seed(2000)
apple_bread_c = function(x, y){
  x_coordinate = c(x)
  y_coordinate = c(y)
  i = 1
  i = 1
  while(i > 0){
    a = runif(1)
    if (a < 0.25){
      x_coordinate[i+1] = x_coordinate[i] + 1
      y_coordinate[i+1] = y_coordinate[i]
      if (y_coordinate[i+1]^2 + x_coordinate[i+1]^2 > 55^2){
        break
      }

    }
    else if(a > 0.25 & a < 0.5){
      x_coordinate[i+1] = x_coordinate[i] - 1
      y_coordinate[i+1] = y_coordinate[i]
      if (y_coordinate[i+1]^2 + x_coordinate[i+1]^2 > 55^2){
        break
      }

    }
    else if(a > 0.5 & a < 0.75){
      x_coordinate[i+1] = x_coordinate[i]
      y_coordinate[i+1] = y_coordinate[i] + 1
      if (y_coordinate[i+1]^2 + x_coordinate[i+1]^2 > 55^2){
        break
      }
    }
    else{
      x_coordinate[i+1] = x_coordinate[i]
      y_coordinate[i+1] = y_coordinate[i] - 1
      if (y_coordinate[i+1]^2 + x_coordinate[i+1]^2 > 55^2){
        break
      }
    }
    i = i + 1
  }
  l1 = length(x_coordinate)
  plot(x_coordinate,y_coordinate)
  z = 1:(length(x_coordinate)-1)
  arrows(x_coordinate[z], y_coordinate[z], x_coordinate[z+1],
         y_coordinate[z+1], angle = 30, length = 0.001,
         col = c("red", "blue", "green", "orange", "darkblue"))
  arrows(0,0,x_coordinate[l1],y_coordinate[l1], col = "darkblue")
  print(c("The flow terminates at"))
  print(c(x_coordinate[l1], y_coordinate[l1]))
  print(c("The number of times the loop ran is:", l1))}
```

```
apple_bread_c(0,0)
```




```
## [1] "The flow terminates at"
## [1]  35 -43
## [1] "The number of times the loop ran is:"
## [2] "3303"
```

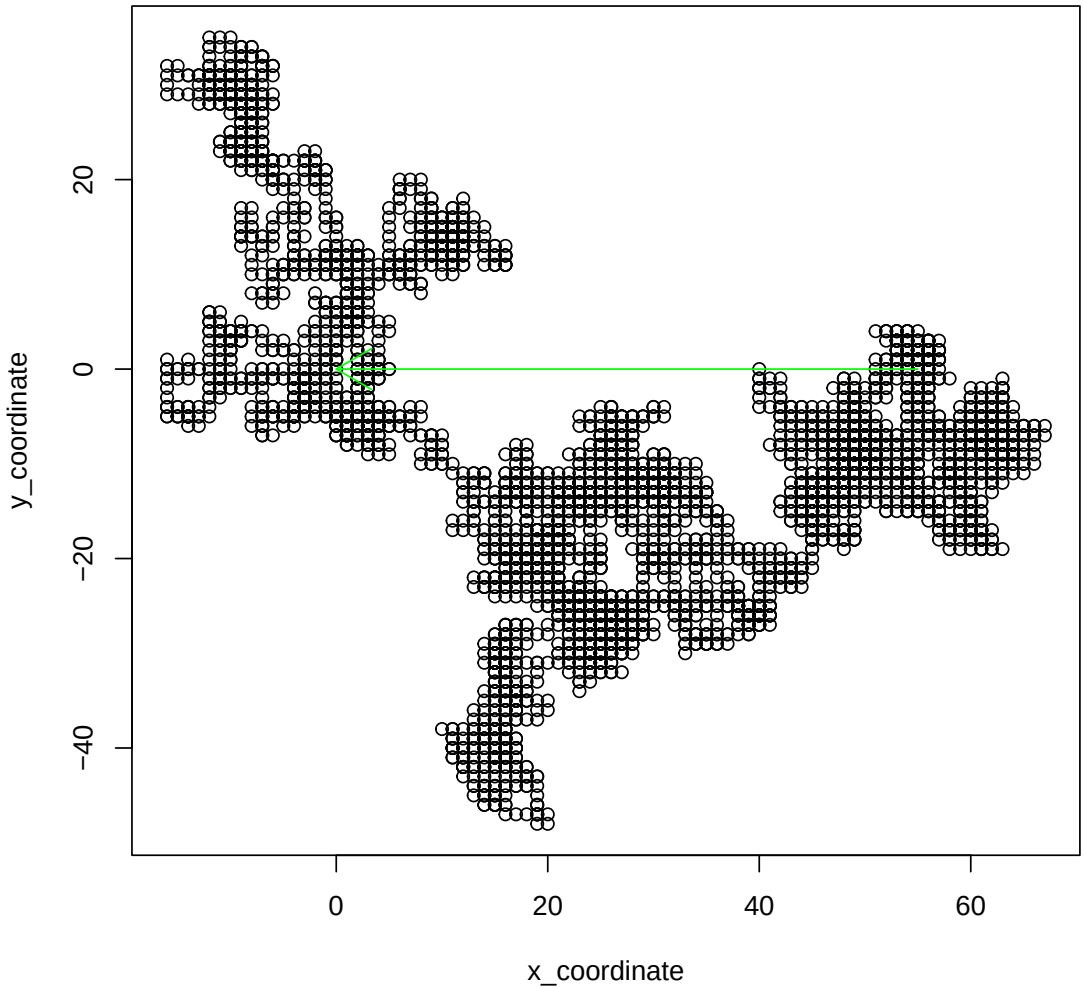
```r
set.seed(2004)
apple_bread_d = function(x, y){
  x_coordinate = c(x)
  y_coordinate = c(y)
  i = 1
  while(i > 0){
    a = runif(1)
    if (a < 0.25){
      x_coordinate[i+1] = x_coordinate[i] + 1
      y_coordinate[i+1] = y_coordinate[i]
      if (x_coordinate[i+1] == 0 & y_coordinate[i+1] == 0){
        break
      }

    }
    else if(a > 0.25 & a < 0.5){
      x_coordinate[i+1] = x_coordinate[i] - 1
      y_coordinate[i+1] = y_coordinate[i]
      if (x_coordinate[i+1] == 0 & y_coordinate[i+1] == 0){
        break
      }

    }
    else if(a > 0.5 & a < 0.75){
      x_coordinate[i+1] = x_coordinate[i]
      y_coordinate[i+1] = y_coordinate[i] + 1
      if (x_coordinate[i+1] == 0 & y_coordinate[i+1] == 0){
        break
      }
    }
    else{
      x_coordinate[i+1] = x_coordinate[i]
      y_coordinate[i+1] = y_coordinate[i] - 1
      if (x_coordinate[i+1] == 0 & y_coordinate[i+1] == 0){
        break
      }
    }
    i = i + 1
  }
  l1 = length(x_coordinate)
  print(c("The number of times the loop ran is:", l1))
  plot(x_coordinate,y_coordinate)
  arrows(x,y,0,0, col = "green")

}
apple_bread_d(55,0)
```

```
## [1] "The number of times the loop ran is:"
## [2] "4906"
```


y_coordinate
20
0
−20
−40
0
20
40
60
x_coordinate